



GrabberBot—Planning and Design

Taking the suggestion from Chapter 13, this bot doesn't need to be quick but it does need to be simple. All we need is for the bot to move to the end of the tunnel, avoid knocking the scroll off its support legs, grab the scroll securely, and return to the tunnel entrance.

With those requirements in mind, let's move forward and develop a solution to this challenge.

GrabberBot Planning and Design

If you'd like to skip ahead to Chapter 15 and take a look at my final solution for the GrabberBot, feel free. Right now, I can imagine numerous bots that could be built to retrieve that scroll, and I'm sure you can, too. Get out a blank Design Journal page and a pen—your goal for this challenge is to develop an alternative to my GrabberBot.

Note There are two blank Design Journal pages left in the back of this book (if you used one each for Chapters 2, 6, and 10). If you need more pages, feel free to make photocopies of the Design Journal page or visit the Source Code area of the Apress Web site to download the page in PDF format.

In the Robot Name box, write **GrabberBot** or create your own name; some other names I considered included **SnatchBot** and **BringItBackBot**. After you've got your bot name picked out, it's time to think about the bot's description.

The Robot Description

A robot like this one—one that has just two or three jobs to perform—might seem fairly simple to describe. With many bots, the description is so simple that you might wonder if it's even worth spending the time writing it out. I've built plenty of bots without any written description; I just knew what I wanted it to do and I started building.

But don't let this bot fool you. There are some things about this bot that are a little tricky and deserve some attention. Take a look at my Robot Description in Figure 14-1. I might have

run into some trouble if I had just started building without considering all the things involved in retrieving the scroll.

DESIGN JOURNAL ☐ ☐ ☐

ROBOT NAME GrabberBot

ROBOT DESCRIPTION

The GrabberBot must move down the tunnel towards the scroll. The bot must stop before it touches the scroll, using a sensor or other method for detecting when to stop. The bot will need to remove the scroll without dropping it and continue to hold the scroll as it moves in reverse, returning to the tunnel entrance. Speed is not important, so the bot doesn't have to move quickly towards the scroll or back to the entrance once it has retrieved the scroll.

Figure 14-1. The GrabberBot Robot Description revealed a couple of tricky items.

Where would I have encountered trouble? Well, when I just start building randomly, I sometimes find that I have to tear it apart and start over because I didn't take an external factor into consideration. With this bot, it is likely that I *would have* thought about a method for the robot to detect the end of the tunnel. But because I would have skipped the other parts of the Design Journal page, I *might not have* considered the limitations on the bot and where to place a sensor so that it didn't interfere with its primary task of grabbing the scroll. That's one of the risks of just snapping pieces together without a plan.

By using the Design Journal page and completing all the sections, I reduce the risk of starting to build and then having to start over when I run into a design that doesn't quite work the way it should.

Let's focus on one of my Robot Description sentences: "The bot will need to remove the scroll without dropping it and continue to hold the scroll as it moves in reverse, returning to the tunnel entrance." This should get you thinking about what's involved for the bot to "hold" the scroll. Will it hold it like a hand would hold an object, with fingers wrapped around the scroll? Could it somehow simply pinch the scroll and pull it away? If so, how could you make certain the scroll didn't come loose during the return trip? Will it come at the scroll from below or from above?

Your Robot Description should force you to start asking your own questions. And it's the answers to these questions that will help you start developing a picture in your mind of what the final bot will look like.

After I completed my Robot Description, I realized that my Task List was going to be very short, too. But a short Robot Description or Task List does not mean the design of your bot will be easy. Sometimes the difficulty is in the building of your bot, sometimes in the programming of it, and sometimes in both. I'll finish up my Design Journal page before I get too excited about the simplicity of the bot's description.

The Task List

My Task List is shown in Figure 14-2. How does it compare to your own?

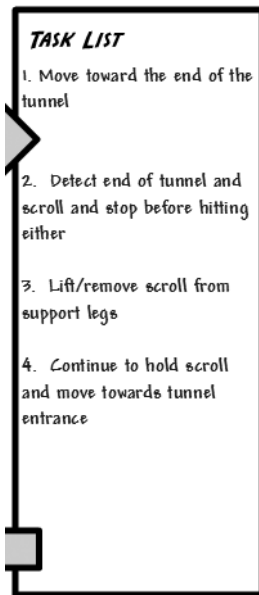


Figure 14-2. *The GrabberBot Task List may be short, but each item is important.*

Let's go through each of the four tasks in detail and develop some ideas. These ideas can be used in the Mindstorm section, as long as they don't violate any of the limitations or constraints in the next section of your Design Journal page.

The first task is "Move toward the end of the tunnel." Simple enough. If we ignore the shape of our bot for now and just concentrate on this task, what attributes of the robot can we consider? Well, one item is the bot's speed. As it moves down the tunnel, do we want it moving fast or slow or somewhere in between? Personally, I don't want to wait forever, so I might plan on sending the bot down the tunnel at a high rate of speed until it gets close to the end of the tunnel. When it nears the scroll, I'd prefer that it slowly approach it, to avoid knocking it off the support legs.

The best part of moving down the tunnel is that the robot needs to move in a straight line only. No turns and no special zigzagging is needed—this bot will go straight to the scroll and then straight back to me.

The next task, "Detect end of tunnel and scroll and stop before hitting either," is a little more complicated. I could use the Sound Sensor and program the bot to stop when I yell "stop!" but this could be less accurate than using one of the other sensors. I'll probably get more accurate results if I choose to use the Touch Sensor or Ultrasonic Sensor; it really depends on whether I want to actually touch the scroll or wall (Touch Sensor) or detect the proximity of the scroll or wall (Ultrasonic Sensor).

The third task is "Lift/remove scroll from support legs." The scroll is sitting about four inches above the tunnel floor on two small support legs. If I accidentally knock the scroll off, I suppose I could build another bot to go down the tunnel and pick it up. But failure isn't an

option for this bot (okay, sometimes it is, but let's think positive). I believe it will be safer to lift the scroll up, coming at it from underneath. If I create something to reach forward (like a hand) I might accidentally push the scroll off the pedestal. I could also create a mechanism that reaches down from above the scroll, but again, there is a slight chance I might cause the scroll to roll forward or backward and fall off the supports.

My final task, "Continue to hold scroll and move towards tunnel entrance," will only work if I've successfully grabbed the scroll and can hold it securely. If my bot has the scroll held securely, I can reverse the direction of the bot and have it move back towards the tunnel entrance. For speed, I'll probably have it move at a medium speed or slower; a fast-moving bot might get to me quicker, but I can afford to let it move slowly if it means not losing the scroll.

By examining my Task List in a little more detail, I've been able to start brainstorming (excuse me, *mindstorming*) about the size and speed of my bot and the components to use in it. But before I continue, I need to examine any constraints my bot might encounter.

Limitations and Constraints

Considering that the GrabberBot is heading down another tunnel, I know of at least two limitations for my bot: size and weight. All of the limitations for my bot are shown in Figure 14-3, in my Limitations/Constraints box.

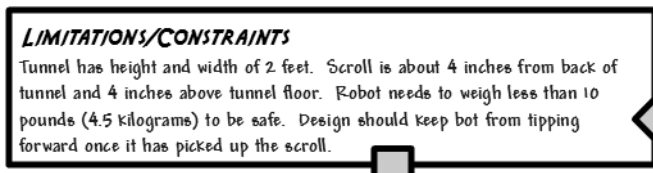


Figure 14-3. The limitations for the GrabberBot must be taken into consideration.

Obviously, my bot must be no taller than two feet and its width should not exceed two feet, either. The bot's weight, on the other hand, could be an issue.

A quick Internet search revealed that a full grown spider monkey averages about 6 to 10 pounds (2.7 to 4.5 kilograms). If I can keep my bot's weight below that number, I should be okay—the pressure plates shouldn't trigger like they would if a grown person were to crawl down the tunnel.

My final constraint is related to the position of the scroll. The scroll is four inches from the rear of the tunnel and about four inches above the tunnel floor. Whatever method I use to pick up the scroll must take those measurements into consideration.

If I can design my bot so that it doesn't violate any of those limitations, everything should be good.

And now, taking into consideration the Robot Description, the short Task List, and the Limitations/Constraints, are you ready to begin mindstorming?

Mindstorm

I know my Task List was short, but the Mindstorm section of my Design Journal page is much longer. Take a look at Figure 14-4.

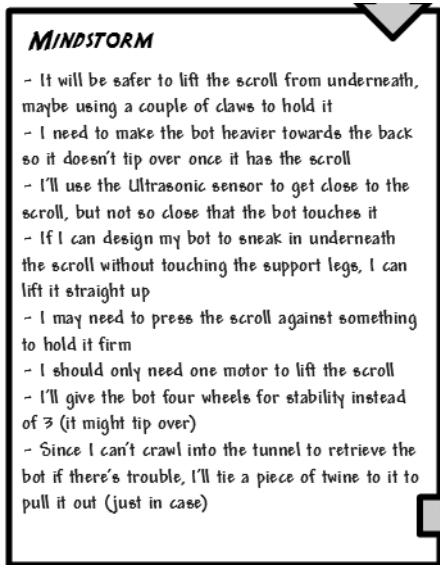


Figure 14-4. The Mindstorm section for my GrabberBot isn't short at all.

The first Mindstorm item, “It will be safer to lift the scroll from underneath, maybe using a couple of claws to hold it,” relates to how my bot will approach the scroll. Earlier in the chapter I mentioned that my bot could reach down and grab the scroll, reach forward and grab it, or reach from underneath. I still believe that the safest approach is to somehow get my bot under the scroll and somehow lift an arm or trap or other mechanism to surround the scroll and lift it directly up off the support legs.

Try this little test: take a couple of tin cans (vegetable soup, for example) and sit them about six or seven inches apart. On top of the cans, place something scroll-shaped—maybe a tube of toothpaste or cookie dough or paper towel roll. Now close your eyes. Have someone direct your hand to the “scroll” using *only* these verbal commands: forward, backward, stop, up, down, left, right, and grab—don't cheat! Don't use your fingers unless you plan on designing your bot with fingers (and I doubt you'll have enough motors to do this). Try different approaches to lifting the “scroll” off the tin cans. Which worked better? For me, an open palm coming in under the “scroll” is the easiest—if I avoid touching the tin cans, all I have to do is raise my hand (“UP!”) and the scroll will sit on my palm.

Another Mindstorm item that I think will be very important is “I'll use the Ultrasonic Sensor to get close to the scroll, but not so close that the bot touches it.” Take a look at Figure 14-5. This is how I'm visualizing approaching the scroll.

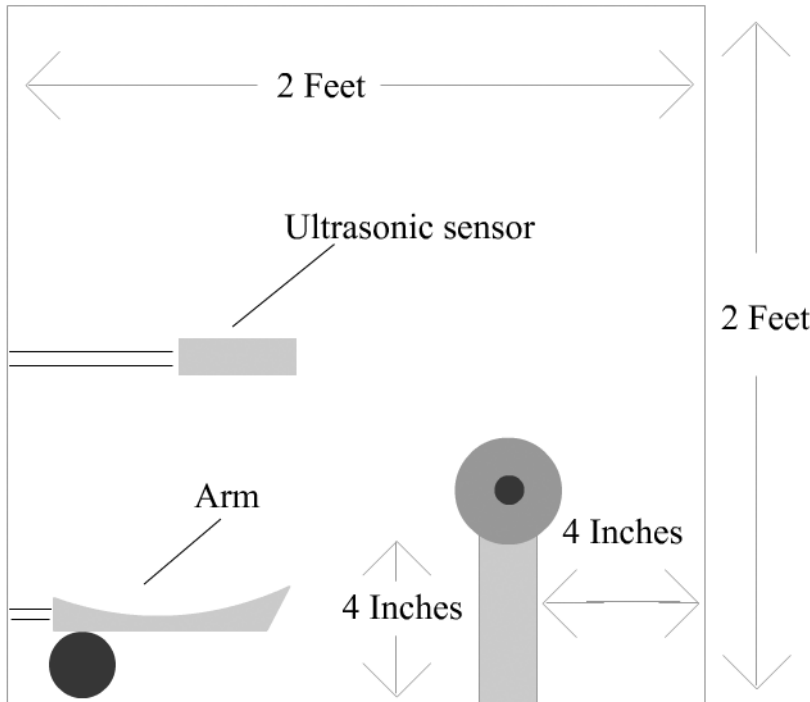


Figure 14-5. *Side view of the bot approaching the scroll*

What I'm considering doing is having the Ultrasonic Sensor above the scroll and some sort of arm mechanism below the scroll. As the bot moves towards the scroll, I'll configure the Ultrasonic Sensor to stop the bot at a certain distance from the wall. That distance will be determined through testing, but the idea is that it will stop the bot when the arm is directly underneath the scroll. The bot stops, the arm lifts up, and the scroll is captured.

Remember, we're just mindstorming right now, so I'm not getting too detailed in my drawing or design just yet. Testing might prove later that this method doesn't work—that's a risk, but that's why I'm spending time thinking about this bot before I start building.

The last Mindstorm item I want to cover is "I may need to press the scroll against something to hold it firm." If I can get the bot to successfully lift the scroll, there is a chance that when the bot begins to move in reverse, the scroll might roll off the arm. What I'd like to do (and I'll test this) is to have the arm lift the scroll up against some beams or other NXT components—this should hold the scroll firmly while the bot moves. Again, I'll have to test this and it might not work.

The rest of my Mindstorm items are fairly self-explanatory. A four-wheel bot will be more stable than a three-wheel bot; a bot that's heavy in the rear won't tip forward when it picks up the scroll; one motor should provide sufficient lifting power; and, something that occurred to me after I designed the SnapShotBot, tying a string to my bot will allow me to pull it out if the bot gets into trouble. If the scroll falls off the support legs, I don't know what will happen, but the best-case scenario for that happening would be to simply build another bot that goes down the tunnel and retrieves the fallen scroll. But let's not let that happen, okay?

Well, we're done with the Mindstorm section of our Design Journal, and there's only one section left before we begin to build.

Sketches

As you saw in Chapter 10 in Figure 10-7, I don't use detailed images of things like sensors and wheels. I don't like to spend time drawing components in detail when I can use a rough sketch to get my point across. I do the same thing when sketching my ideas for the shape of my bots. While completing my Design Journal page, I began to build some rough pictures in my mind of what I want my bot to look like. So take a look at Figure 14-6 and remember not to laugh too loud or you'll disturb your neighbors.

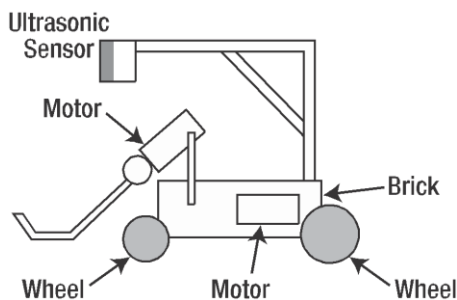


Figure 14-6. Once again, basic shapes are used to represent parts of my GrabberBot.

Again, I'm not drawing every sensor, wheel, beam, and connector piece. I just want to use basic shapes to represent those items. The brick is easy—rectangle. Wheels are circles. Motors are still weird looking, but I label them so I remember what they are.

When I'm done, I've got a good idea of where to start building. And that's exactly what I'm ready to do. Let's move on to Chapter 15 to get the GrabberBot built and ready to retrieve that scroll.